SUPPLEMENTARY MATERIALS

USENIX Systems Administration (LISA VII) Conference

November 1-5, 1993
Monterey, California

For additional copies of these proceedings contact

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710 USA
Telephone: 510-528-8649

The price is $25 for members and $33 for nonmembers.

Outside the U.S.A. and Canada, please add
$12 per copy for postage (via air printed matter).

Past USENIX Large Installation Systems Administration Workshop
and Conference Proceedings (price: member/nonmember)

Large Installation Systems Admin. I Workshop	1987	Phildelphia, PA	$4/$4
Large Installation Systems Admin. II Workshop	1988	Monterey, CA	$8/$8
Large Installation Systems Admin. III Workshop	1989	Austin, TX	$13/$13
Large Installation Systems Admin. IV Conference	1990	Colorado Spgs, CO	$15/$18
Large Installation Systems Admin. V Conference	1991	San Diego, CA	$20/$23
Large Installation Systems Admin. VI Conference	1992	Long Beach, CA	$23/$30

Outside the U.S.A. and Canada, please add $8
per copy for postage (via air printed matter).

Printed in the United States of America on 50% recycled paper, 10-15% post consumer waste.

USENIX Association

Supplementary Material of the Seventh Systems Administration Conference (LISA VII)

November 1-5, 1993
Monterey, CA, USA

Managing an Ever-Changing User Base

John E. Miller – Lewis & Clark College

ABSTRACT

While it is relatively easy to add numbers of different kinds of new users to a unix system, it is not so easy to selectively remove them according to some liberal policy at the appropriate time. This paper explains our strategy for managing such accounts through a system of programs loosely coupled to our institutional personnel and student data base. A menu allows the bulk of the processing to be handled clerically, freeing us to develop higher-level account management tools.

Our package, including perl scripts, c-programs, forms, policies, and procedures, is available via anonymous FTP.

Background

While Lewis & Clark is the largest private liberal arts college in Oregon, it is small enough to be served by a single NIS domain, even including our graduate and law schools (1,500-2,000 users altogether). We restrict the use of some of our workstations through netgroups. Otherwise, any user can log in anywhere.

L&C has one main academic system (a Sun 4/490 running SunOS 4.1.2), a dozen workstations, and hundreds of Macintoshes and PC's. We run the Pine mail interface, IMAP clients, rn, tin, nntp newsreaders, gopher, CAP, and other services off the main system. Computer Science courses are taught using our DECstation Laboratory which is connected to the main system via NIS and NFS. The Sun also does some intensive computing for a few users.

Likewise, we have one administrative system (a Prime 5370 running COLLEAGUE under PRIMOS) that provides data base services for the college with its own set of users. Administrative staff use the academic system for mail and news, however, and have unix passwords for that purpose.

Introduction

At most sites, the set of unix accounts is not static. At some sites it is more like running an airline system than a computer system – users coming and going all the time. Unfortunately, if old users are not removed, their accumulated files will kill you one way or another, or their password will end up in the wrong hands.

How do you know when the time has come to remove a user? It seems like a waste to build and maintain a data base of user information just to answer this question when there is probably such a data base just across your network.

Historically, it has been the responsibility of system managers to authorize and track users on corporate and collegiate systems. Typically, no institutional policy insured that these managers ever found out that certain users should no longer be welcomed. It seems that this responsibility must be shared between the system managers and the institution or corporation, whether codified in policy or just proceduralized.

Assertions

We have made some pretty pathetic attempts to track users by ourselves. For example, when we switched from VMS to unix, we gathered things like users' campus mail box numbers (so that we could send paper mail to users if warranted) and put them in the old GECOS field of `/etc/passwd`. The campus post office then proceeded to install new mail boxes and reassign box numbers! Likewise, the expected year-of-graduation turned out to be fairly worthless information. So, from experience, we have made the following assertions:

- New users arrive chaotically. If you create 100's of accounts for first-year students and new-hires in anticipation, you will have many unclaimed pieces of paper and unused accounts fattening your password file. Therefore, create proper new accounts only on demand.
- College students graduate after various lengths of time, or not at all. Faculty and staff leave chaotically.
- Each individual should retain their login for the duration of their stay at the college, whether faculty, student, or staff member. (We find that over half of the students who get logins for a class continue to use mail and other services after that course is over.)
- If a graduating student wishes to stay on as an alum user, they should be able to retain their identity.
- Users may take a leave of absence, go overseas, or otherwise appear inactive.
- Only the registrar and personnel know for sure who is enrolled, employed, about to graduate, has been terminated, and so on. We need to get information from them.

- Much of the work of maintenance is mundane, and should be on a secure menu for use by a clerk or coordinator.
- It would be useful to be able to trace the history of an account.

Solution: Use "Their" Data Base

We met with the administrative information systems director to explore ways that we could validate our user base against the institutional data base. To our surprise, she suggested that if login requests were first passed through the administrative system, users could be validated and tracked outside of unix using enrollment and employment status. The idea was so compelling (and blasphemous) that we designed our new system around it. The director expanded COLLEAGUE's data tree, created a screen for data entry, and did whatever things were needed on the administrative system to deliver the service we needed. Hereafter the subset of the administrative system that deals with unix logins will be referred to as "ULS", while the academic system will be called "UNIX". The features of our combined system are:

- A screen on ULS processes account request forms. The same program generates a unique login-key for each new record. All the records are kept in LC.UNIX.LOGINS, a branch of the administrative data base.
- An end-of-day transfer moves new records to UNIX where the corresponding accounts are generated overnight.
- ULS keeps a list of words such as "root" that cannot be used for new user's logins in LC.UNIX.RESERVED, an auxiliary code file. This list includes all the logins on UNIX that are not in LC.UNIX.LOGINS, such as our guest accounts, logins from `/etc/passwd`, and selected aliases from `/etc/aliases`.
- Accounts can be removed by a variety of methods, some based on information from ULS, some from informal channels, some from our own observations of inactivity, policy violations, and so on. (There is no one criteria for removal).
- UNIX keeps only an ID number as a cross check. We have not had to use it much in practice, but it is there should we need to access the administrative data base driven from the UNIX side instead of the usual selective process originating on the administrative side.

The benefits of doing business this way are undeniable:

- Users are "authenticated" before we put them into our system.
- Unix login names are available for publication in our faculty-staff and student directories.
- Any user's current home address and status is available to us at no cost.
- ULS provides candidates for removal.
- If we want data for a nameserver or other application, we can request it from ULS periodically.
- From our viewpoint, the institutional data base maintains itself, i.e., we do no work to maintain it.

Daily Operation

Throughout the day, non-users apply for accounts by completing a UNIX ACCOUNT REQUEST FORM. The non-user supplies their ID, name, and a phone number where they can be reached if we have a problem processing their request. The non-user also indicates whether they are faculty, staff, student, or an alum. The non-user must present an ID card so that we can verify their ID number. The form also contains a responsible use policy. The user signs the request under agreement that they will abide by all our unix policies. (Details of our policies and procedures are included in our distribution.)

Each form is handled by a custom screen in ULS, usually toward the end of the day. ULS checks that the applicant is enrolled or employed, and then creates a unique 3 to 8 character key for the ULS record by using a series of name-forming rules. (Our account coordinator has access to this part of the data base so that our office can perform the data entry).

After we go home, ULS retrieves all newly created records, with login, first and last names, login, group code, and ID and exports them to UNIX via ftp.

UNIX then runs an account creation script via cron and notifies our account coordinator and myself via mail. The following workday our coordinator prints account notification sheets for the new accounts and gives them to our front desk for pick up. Applicants return to claim new accounts with their ID.

Note that accounts are created overnight, not instantly on demand. We find this cools things off and puts a reasonable level of expectation on this service. "It will be ready in the morning".

Password changes and other requests can be handled the same day because they affect only UNIX. Name and group changes can be made during the day even though ULS records must be modified.

An account for a guest of the college can be created on demand. See the section **New Users** below.

Groups

Before we describe the actions of our co-system in detail, we need to define our groups. Our local culture lends itself to the following set of groups:

- Faculty: full and part-time faculty, librarians, and other academic support staff.
- Staff: all other employees.
- Student: currently-enrolled students. We make no distinction between law, grad, and undergrad students, but we could.
- Alum: graduates of the college. Some come onto the system for the first time as alums, while others "graduate" into the alum group (See the section **Changing a User's Group** ahead).
- Guest: people who are not in our institutional data base, such as spouses, children of faculty or staff, colleagues of faculty members, et cetera.
- Organization & Department. These are so static that they are managed manually and merit no further discussion.

We have these groups so that we can assign home directories to different file systems, and so that we can distinguish one from the other for service delivery (e.g., Alums do not necessarily get the same level of support).

Actions

Any action affecting our password file is guarded by a lock to ensure that no one is editing it, or using the accounts menu from another station. Since we are running NIS, our `acctlock` script kills the password daemon and restarts it so the user cannot change the file either. (`yppasswdd` evidently can't just be signaled to disallow password changes). After the NIS map is made, the lock is removed and the daemon is restarted to allow user-initiated password changes.

New Users

As described in the **Daily Operation** section, logins are actually created as a key in a branch of the administrative data base. Our nightly job, `autoacp`, takes the batch transferred from administrative system and invokes our `acp` (account creation program) to create passwd entries, directories, and some "dot files" for each user. `acp` creates an `acp.log` file with the plain text passwords for later printing. `acp` also logs each creation in the accounts logfile.

We allow a limited number of guests on our system. Since guests are not typically found in ULS, they can't be created through the above process. Instead, we run a `newguest` script (from a menu) that gathers the name and desired login and then invokes `acp` to process the request. `newguest` also mails a reminder to the account coordinator: "Please add the following names to our LC.UNIX.RESERVED code file" so that the name(s) will be excluded from use as a key in LC.UNIX.LOGINS.

`printsheets` (from the same menu) takes the logfile created by `acp` and prints account notification sheets. Each sheet has a user's login and password. The bulk of the notification sheet consists of tips and reminders. Once the sheets are printed successfully, the `acp.log` file is removed.

Removing Users

Our `remove` script removes passwd entries, home directories, mail spool files, and the entry from the student-ID-vs-login file. It then sends mail to the coordinator with a list of the logins removed, along with their ID numbers for removal from ULS. Lastly, `remove` records each removal in the logfile. (`remove` currently does not remove names from any aliases in `/etc/aliases`).

Our removal strategy is, of course, to determine which accounts should no longer be on our system, then put the login names into a file and run `remove` on them. We do not take removals lightly just yet, so this is not a menu item.

`remove` is driven by lists of logins. We have a filter that can pick out the login in the first field no matter what the separator, so that we can come up with lines from passwd, uploaded lists from ULS, output from various commands, whatever, and use them to drive the removal.

Easily removed are: unclaimed accounts, accounts picked up but not accessed, and expired guest accounts. (We give the unclaimed and unused account holders a reminder through campus mail before removing their accounts. We contact guest sponsors before removing guests.)

Capitalizing on the information in the administrative data base, we can periodically request a list of terminated employees and "dropouts" who had unix logins. Before removing these users we retire the previous night's backup tape and then run `remove` on the list.

We have a program `ll` that selects logins on the basis of the owner's GID and the number of days since they last logged in. This allows us to remove inactive accounts of any kind.

Changing a User's Group

The problem of when and how to remove graduating seniors has been complicated by our policy of granting alum accounts. This last year we seem to have hit on a fair and effective procedure.

Prior to graduation, we request a list of all graduating unix logins (!) and a set of mailing labels for the graduates. We then send out a paper letter to the effect that their unix login will be preserved, but that all their files will be removed at a specific date and time. We advise them that it is their responsibility to download any files that they want to keep. We tell them that if their mail inbox is not purged of month-old and older messages, we will remove the whole inbox. We also tell them not to worry if they do not intend to use the account, since it will be removed when it appears inactive.

At the appointed time, we run our `graduate` script with Pomp & Circumstance playing in the background. New home directories are created. Selected dot-files are moved over to preserve newsgroup selections, addressbooks, and aliases. Their old home is removed. Their mail inbox is removed if it contains any month-old messages. Finally, the passwd file is updated with their new GID and HOME. We restrict all logins during this time.

The main advantages of this method are that we get to reclaim the disk space, the users get to keep their logins, and continuity of service is not disrupted unreasonably. One side-effect is that their inbox will appear to have been read if not removed.

We also have a `change-group` script for changing a student or alum into a staff member, and so on.

Removing "Special" Users

Any time an employee leaves suddenly or a student gets out of line, we can remove them on the spot with `remove`. (Recall that `remove` notifies the accounts coordinator to remove the login from ULS). We can also flag any ULS record so that a user cannot quietly request a new account at a later time.

Reconciliation

In order for this co-system to work properly, there must be a consistency between their respective bases:

- LC.UNIX.LOGINS under ULS must be consistent with the UNIX passwd file (minus guests, departments, and organizations).
- LC.UNIX.RESERVED under ULS must be consistent with our generable `lc.unix.reserved` file, containing our guest, department, and organization accounts, logins from `/etc/passwd`, and selected aliases from `/etc/aliases`.
- It is not necessary, but nice if the shadow ID file (login vs L&C ID) is consistent with the passwd file. We use it if there is ever any doubt about name collisions.

  Because the transfer of information from UNIX to ULS is via mail and human operator, inconsistencies can develop. Periodically, we generate files from each of the above sources, find the differences and iron them out. After we run the system another year, we'll have a better idea of how well this works. If UNIX could communicate its actions directly to ULS, we could achieve 100% consistency.

Wrinkles & Puzzles

Basing activity on `lastlog` is incorrect. What about imap, nntp, X, and other things that don't touch `lastlog`? We need a file named "`lastauth`", that gives the last time a user was authenticated.

How would what we are doing scale to a larger organization?

A staff member might come on as a student and then graduate. A student in the payroll data base (because of temporary job) can be terminated but still be a student.

The list of graduates contains some almost-but-not-quite graduates. They will holler when they get the letter.

"Never logged in" could mean that the account was just created yesterday. This alone is not a cause for removal without considering the date created.

Several new professors invariably need their login in June, for publication in a professional directory, but will have nothing official in the administrative system till the end of August. So we create them as a guest, change to faculty later and make changes from RESERVED to LOGINS in ULS.

Adjunct faculty appear to be terminated at the end of every part-time contract with the college. We keep a list of adjuncts so that we can handle personally.

What we really want is a system-wide authentication scheme for all our network resources such as AppleShare, Pacer Forum, laser printing, E-mail, News, Novell, et cetera. Otherwise, some of these other things require separate user bases.

System Startup

To start this system, we had to get our active set of logins into the COLLEAGUE data base. Our administrative computing shop did the programming to allow maintenance of the data. Likewise, our "reserved words" needed a screen to maintain them as an auxiliary code file. We transferred the daily batches to unix via kermit until we figured out how to get ftp working from the PRIMOS batch.

After that, it is a matter of extracting lists of logins for removal.

Conclusion

Every computerized organization has an administrative data base containing information about currently active employees or students – the payroll roster for businesses, or registration rolls for schools. The user accounting system should make as much use of these pre-existing databases as possible, and rely on the administrative data base for correctness.

This method is good because it includes the institutional data base in the loop, capturing the data at point of entry. Instead of us system managers trying to get information on our users from the institution, the institution can tell us things about our users. The integration of unix login names with institutional data base benefits both ‘‘sides’’.

This method requires cooperation with your administrative shop because you are not simply getting lists of people in classes who need accounts for the term. The administrative side will need to remove logins when you do, etc. Maintenance privileges on the administrative system makes things easier.

Future Work

1. Automate ULS removals from lists we provide.
2. Fine tune and automate all requests that can be made against the institutional data base. The biggest part of this is knowing the best ‘‘data’’ time to make each kind of request.
3. Fine tune double-checking of lists that are generated for removal, i.e., what do we do if a user is up for removal, but appears to be more active than expected? These situations need be handled on a case-by-case basis, since we can’t be sure that the user shouldn’t just be left on the system in order to avoid the hassle of restoring their files from backups.
4. In the near future, users will be connecting to services via desktop clients, we must find an alternative to using `lastlog` to determine account inactivity.
5. System V.

Availability

Everything we have is available via ftp lclark.edu, pub/accounts.tar.Z, including PICT and PostScript versions of our forms and the TeX version of this paper. Other useful scripts, not relevant to this paper, are included. If your site is running Colleague, you can have INFOBASIC sources as well.

Acknowledgements

I am grateful to David Webster for all his constructive arguments leading to this system, and for help tracking all our users. David, Jon Herlocker, and Joshua Gerth did the programming. Thanks to Karen King for her insight on using the institutional data base rather than trying to maintain our own.

Author Information

John Miller got his BS in Information Science from Washington State University in 1972, taught computer science from 1972 to 1982, and has been a unix system manager and scientific consultant from 1982 to present. His mail address is: Academic Technologies, Lewis & Clark College, 0615 SW Palatine Hill Road, Portland, OR 97219. John can be reached via E-mail at miller@lclark.edu.

The Myers-Briggs Type Indicator: An Interpersonal Tool for System Administrators

Betty Jacob – PRC, Inc.
Nancy Shoemaker

ABSTRACT

Tools to automate, improve, and provide insights into the technical environment of system administrators are widely available. This paper focuses, in contrast, on a tool to improve the interpersonal environment within which system administrators work. SA's often become focal points for interpersonal communications, and they need to handle this aspect of their jobs well in order to fully realize technical success. This paper presents a tool which is widely used but which may not have been introduced to many system administrators.

The Myers-Briggs Type Indicator <3> describes interpersonal differences and provides a framework for problem solving and conflict resolution. This paper introduces the four dimensions of the MBTI, provides information on likely patterns of types represented by SA's and contrasts this with managers and users. We illustrate the four dimensions of the MBTI with applications from system administration. We provide some discussion of the limitations of the MBTI, and give practical examples of its use in an SA setting.

Introduction

The Myers-Briggs Type Indicator (MBTI) is one of today's most widely used personality assessment tools. Used properly, it can provide system administrators a framework for understanding themselves, their co-workers, and the user community.

There are 16 personality types in the Myers-Briggs taxonomy. Types are not right or wrong: they are simply different from one another. Type theory provides a non- judgmental way to describe and discuss differences. Once there is a language to describe differences, more effective communication can result.

As will be discussed later in this paper, the MBTI is not meant to be a panacea. We do not believe that the MBTI alone can explain all personality preferences, differences, strengths, and conflicts. We do believe that it is an easily learned tool that can help provide a greater understanding of the human dimension.

Overview of Characteristics of Types

The MBTI is based on the premise that each person has a **preferred style** of operating. This is critical: The MBTI is not based on skills or aptitudes. The same skills can be exhibited in different styles, and people can occasionally exhibit a style other than the one they prefer. Since the MBTI is strictly a self-assessment, a person taking the inventory is essentially categorizing his or her own preferences. This preferred method of operating can be measured in four different dimensions, discussed below.

Source and Direction of Energy: Extraverts[1] and Introverts

The Extravert/Introvert (E/I) dimension deals with a person's source of energy.

- Extraverts get their energy from outside themselves
- Introverts get their energy from inside themselves.

Extraverts are often characterized as gregarious, talkative, and expressive. They have an outer (external) focus, and often "speak to think." Introverts, in contrast, have an inner (internal) focus, and are often described as reflective, private, introspective. They like to "think to speak."

The E/I dimension also deals with where a person's energy flows out. Extraverts typically direct their energy outside themselves; introverts inside themselves.

Keirsey and Bates [1, p. 25] estimate that about 75% of the population is Extraverted, while only 25% are introverted. For college graduates, the distribution may be nearer 50-50 [2, pp. 14-15].

The Perceiving Function: Sensing and iNtuition

The perceiving function deals with how a person gathers information. If a person takes in data in a factual, realistic and literal fashion, using the five

[1] Dictionaries prefer the spelling extravert, but C.G. Jung preferred extravert and the type theory literature follows his practice.

senses, he or she is probably a Sensor. An iNtuitor, in contrast, takes in information in a more theoretical, abstract, and conceptual manner. This is the S/N function: iNtuitors are designated by the letter N to avoid confusion between this dimension and the Introverted dimension.

Most people with a Sensing preference describe themselves as practical; iNtuitors often describe themselves as innovative. INtuitors search for possibilities, relationships and meaning; and often focus on the future. Sensors are down to earth, realistic, and look to the past and the present.

Estimates are that 75% of the general population in America have a Sensing preference while 25% have a preference for iNtuition [1]. Other estimates of college graduates [2] give a distribution closer to 50-50, but note that even relatively large groups can have an overwhelming majority of those with S or N tendencies [3].

The Decision Function: Thinking and Feeling

This dimension deals with how people make decisions. People with a Thinking preference, make decisions in an objective and impersonal fashion. Individuals with a Feeling preference, make choices based on more subjective and personal factors.

All people are capable of using both types of decision making skills. Feelers **do** think, and think quite well; Thinkers **do** feel. The only distinction is a person's preference.

The population is relatively evenly distributed between Thinkers and Feelers [1]. More men tend to be Thinkers, while more women prefer the Feeling function.

Relating to the Outer World: Judging and Perceiving

Persons who prefer the Judging function, generally choose closure over keeping options open. Perceivers are more spontaneous and flexible, and are typically amenable to looking at new options, information, and possibilities that might change their plans. A Judging person tends to be very structured, organized, and focused; a Perceiving person is more adaptable and spontaneous.

Estimates, [1], are that about 50-60% of the population prefers Judging while the rest prefer the Perceiving function. One sample of college graduates, [2], rates them as even more likely to favor the Judging preference. Some samples reported in [3] (all of size at least 50) are more than 90% Judging, but none are more than 70% Perceiving.

Illustrations from System Administration

In this section, we give further examples of the four dimensions of the MBTI with reference to life in system administration. We emphasize that a good system administrator can exhibit any of the eight preferences. Our goal here is to make the above discussion more concrete.

Extravert/Introvert: Approaches to Meetings

Few things may generate as many negative comments in the workplace as the dreaded meeting:

> I didn't get anything done today (all week? all month?) since all the time was spent in meetings.
>
> That meeting was a complete waste of time.

A system administrator's role often ensures that there will be several meetings during the week: meetings with management, meetings with users, meetings with vendors, meetings with other members of the technical staff.

Granted, organizations that ignore principles of well run meetings (see, for example, [4]) can waste everyone's time. But some meetings, particularly less formal ones, can be made more productive (or at least less frustrating) if looked at through the lens of the Myers- Briggs Extravert/Introvert dimension.

Remember that Extraverts get their energy from others and enjoy generating ideas with a group. They may present tentative ideas before they are fully worked out, and use the experience of talking through the idea to refine it. For an Introvert, this sounds like rambling and fuzzy thinking. If, however, the Introverts can suspend judgment until a more fully formed idea appears, they can contribute to the group effort by performing the role of a sounding board. The Extraverts can help by giving signals ("I'm just brainstorming here", "Let's try out this idea") when they are talking without a clear destination in mind.

Introverts, on the other hand, prefer to generate their ideas in private and tend to be reserved in group situations. They may appear to Extraverts to be disconnected from the meeting. They may not formulate their reactions until after the meeting has adjourned. There is a risk that their contributions will be overlooked, and recognizing this risk before the meeting is important for both the Introverts and Extraverts. If the Introverts' contributions will be required, an early, clear agenda may be invaluable in allowing the Introverts the private time for preparation. Issues that arise on the spur of the moment might be better handled with a "pre-meeting" to allow the Extraverts to brainstorm the issues followed by an intermission to allow the Introverts to formulate their responses. During the meeting, an awareness of the Introverts' tendency to hold back can make the Extraverts more willing to explicitly yield the floor.

Sensing/iNtuitive: Approaches to Information on System Usage

Recall that the Sensing and iNtuitive functions represent the two preferred modes of gathering information. Let's go through an example of a system administrator doing initial research to justify a system upgrade.

The Sensing SA may prepare data showing the history of disk space usage, CPU usage, memory usage, network bandwidth and the time to complete a typical "job". The iNtuitive SA may note that the number of calls on "slow response time" has increased, recognize that a major new application appears more often at the top of ps listings, and feel that the number of complaints of applications crashing for want of swap space has become unacceptable.

All of this information is useful: both "just the facts" and "the bigger picture" will be helpful in convincing either Sensing or iNtuitive managers to accept the SA's recommendation. The MBTI implication for SA in this dimension is to be careful to gather information using **both** modes – if necessary, enlist an accomplice with the other preference.

Thinking/Feeling: Approaches to a Request for Time Off

Differences in the Thinking/Feeling dimension are apt to be easy to spot in personnel decisions. Consider being the leader of a team charged with converting a production system from one hardware platform to another. Everyone's been working hard for three months and cutover is a week away. A member of the team asks for two days off to spend time with a relative who has suddenly dropped into town. What do you do? Your Thinking or Feeling preference does not determine your decision, but it does determine the criteria you consider in reaching that decision.

If your preference is Thinking, you may weigh the effect of the person's two days off on the schedule, on the workload of the other team members, on the risks associated with that person's particular expertise being unavailable, and on the amount of time off that person has taken in the past. If your preference is Feeling, you may consider how one person's absence will affect the others' morale, or how the employee will feel if allowed or not allowed to spend time with the relative. Weighing these factors may lead you to either a "yes" or a "no" decision, or to a compromise position. The lesson of the MBTI is to recognize your tendency to prefer the Thinking or the Feeling mode, and, particularly for critical decisions, be sure that you have considered factors from the side that you do not prefer.

Perceiving/Judging: Approaches to Installing a New Package

With this dimension, there is a good chance for a system administrator's type to contrast strongly with the rest of the organization. While Judgers thrive on order and work well with schedules, Perceivers rough out plans as they wait to see what new information will present itself. A Perceiving tendency, then, can be an asset in the many system administration positions where it is difficult to predict what issues might arise. This tentativeness in plans can seem like disorganization to the surrounding Judgers.

Consider the task of installing a new software package. An SA with a Perceiving style would schedule the installation and expect it to take perhaps two hours. As the installation starts, she realizes that the disk for local software will be filled in the next couple of months, so she starts rearranging the file systems. This leads to the installation of the new disk drive that has been sitting in the storeroom for a few days. And this can lead to updating

ISTJ Life's Natural Organizers	ISFJ Committed to Getting the Job Done	INFJ An Inspiring Leader and Follower	INTJ Life's Independent Thinkers
ISTP Just Do It	ISFP Actions Speak Louder Than Words	INFP Making Life Kinder and Gentler	INTP Life's Problem Solvers
ESTP Making the Most of the Moment	ESFP Let's Make Work Fun	ENFP People are the Product	ENTP Progress is the Product
ESTJ Life's Natural Administrators	ESFJ Everyone's Trusted Friend	ENFJ Smooth-Talking Persuaders	ENTJ Life's Natural Leaders

Figure 1: The Type Table

the storeroom inventory. And on and on. Two things are apparent:

- When managing such a complex TODO list, a Perceiver needs to be careful that tasks are completed in a correct order and that tasks do not get "lost".
- Never give anyone the original 2 hour estimate!

The Judger, on the other hand, would tend to see the work that the Perceiver accomplished as discrete tasks, separately scheduled, and would be more likely to complete the "software installation" in a time frame close to the original estimate.

The Perceiving and Judging dimension is the one where the second author has seen the strongest contrast between her style as a working system administrator and the style of the organization surrounding her. She is a confirmed P, and has invariably reported to managers who exhibited J preferences. Learning about the difference in this dimension has helped explain the success of tactics that did work with certain managers. For instance, trying to prepare a detailed timeline for projects seemed hopeless: new information changed the schedule so that things were late before they started. On the other hand, providing higher level lists of work in progress satisfied one manager's need for a "schedule", but could be made flexible enough to be useful in keeping the system administration work on track. Understanding and acknowledging the

Temperament	4 MBTI Types	Primary Concern	Style
NF: Idealist	ENFJ, INFJ, ENFP, INFP	Identity, self-realization of higher good	Catalysts
NT: Rationals	ENTJ, INTJ, ENTP, INTP	Knowledge and competence	Visionary
SJ: Guardians	ESTJ, ISTJ, ESFJ, ISFJ	Belonging; the preservation of resources	Stabilizer and Traditionalist
SP: Artisans	ESTP, ISTP, ESFP, ISFP	Variation and Spontaneity; Action	Trouble Shooter and Negotiator

Figure 2: The Four Temperaments

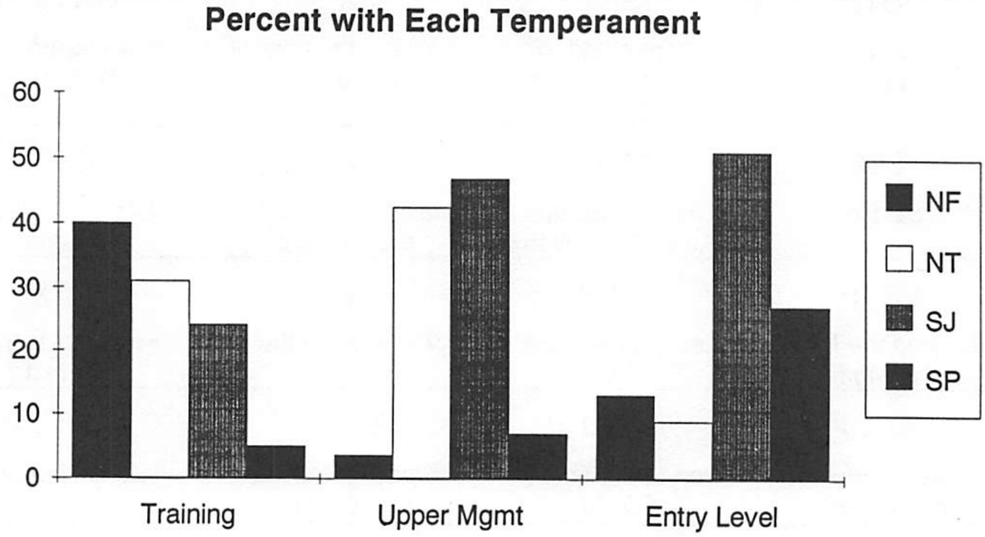

Figure 3: Sample Temperaments in Business

Judger's need for schedules improved the SA's motivation to give priority to the task of preparing the lists. Understanding the benefits of approaching an SA job as a Perceiver allowed the manager to accept the less formal schedules.

Working together

Now that we have discussed the eight functions, let's move on to discuss how the preferences interact.

The 16 Types

It is a tenet of "Typewatching" that knowing your own or others' four preferences, e.g., Extraverted, Sensing, Feeling, Judger (ESFJ) will give information that you can use to improve communications. The four dimensions are independent, so there are 16 possible combinations. In the MBTI literature, these sixteen "types" are shown in a four by four matrix, the type table.

A complete type table is shown in Figure 1, giving the 16 combinations and a short, simplistic description of each type. Kroeger and Thuesen, [5] or [6], provide a description of the 16 types that is much more detailed than we are able to give here.

How can this list of 16 types help you? First of all, it can be a "hook" to help you know yourself by understanding your type and your preferences. If you already know your type, validate it through self-examination. If you do not know your type, review the second and third section, and try to ascertain your 4-letter designation. Compare it to the description in Figure 1. You will be given an opportunity to take the MBTI at the LISA VII conference.

As you go through the process of type validation remember several key points. There are no "wrong" types, nor are certain types "better" than others. Taken individually, each of the four "letters" indicates your preference for operating in a particular domain; together, the four letter designations are indicative of characteristics of a group of people with similar preferences. These characteristics are generalizations only and in no way undermine the uniqueness of the individual. Once you know and validate your type, you can begin to extend your type exploration to examining other people, their preferences, and their types. The insights you gain into how others think and act can be invaluable.

The Four Temperaments

Related to the MBTI is David Keirsey's work on temperaments, [1]. Sometimes referred to as a "Shortcut," the temperaments provide a way to group the MBTI types into four "temperaments" that can simplify looking at interrelationships between types. Figure 2 illustrates the temperaments and their characteristics. In the remainder of this section we will discuss the distribution of temperaments and how different people of different temperaments can work together.

While we emphasize that the types apply to individuals, it can be useful to look at different patterns of the distribution of temperaments in groups. Kroeger and Thuesen [5] give summary type tables for a number of different groups. Choosing three of their groups – trainers and educational specialists, upper managers, and entry level employees – we show the distribution of temperaments in Figure 3.

Differences in styles are also apparent in computer professionals. Selecting the data from three of the tables in the *CAPT-MBTI Atlas*, [3] – computer and peripheral equipment operators, computer programmers, and computer systems analysts and support representatives - we get the summary table of temperaments shown in Figure 4. Given the data for other computer professionals, we would expect to find all four temperaments (indeed, all 16 types)

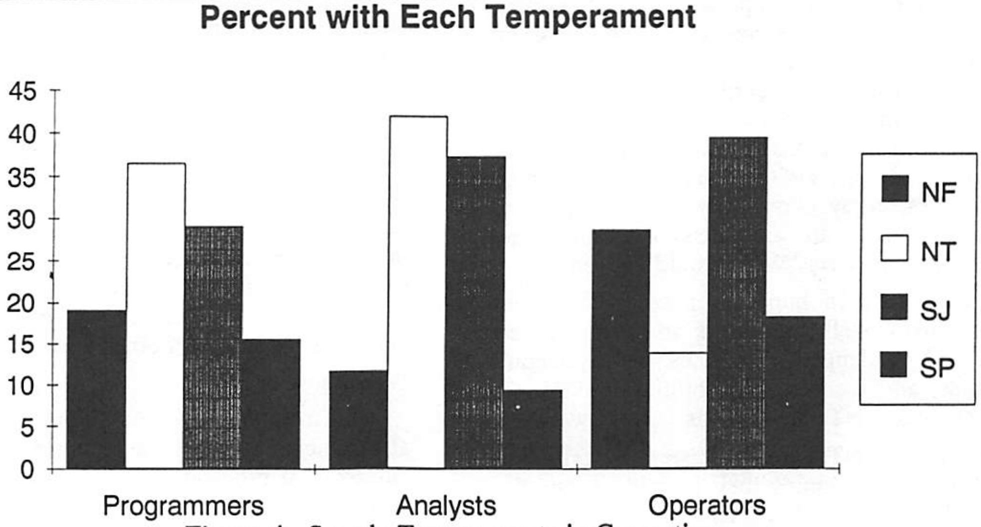

Figure 4: Sample Temperaments in Computing

represented in the population of system administrators. Again, the MBTI says nothing about technical skills and aptitudes, but a working style can, perhaps, be more or less effective in relation to a particular environment. Contrast the environment of a well-established production network with that of a startup venture going through rapid change. While each system administrator may exercise many of the same skills, an SJ/Stabilizer **style** may be better accepted in the production environment while an NT/Visionary may better match the startup. On the other hand, since the MBTI type is a **preference**, recognizing a variation in styles may allow a system administrator to modify his or her behavior to improve communications with management and users. If we go beyond considering features of the work environment and look at any system administration job in greater detail, we often see that one person is now handling tasks that a 1970's "computer center" would have assigned to many different people (with, perhaps, different temperaments). The context switch from systems programmer to user consultant, can be a shift of **style**, not just skills. Looking at the shift in terms of MBTI preferences can add a new dimension to understanding the challenges of the system administrator role.

Knowledge of what types are "typical" of certain groups or occupations may, in the absence of any other information, help you prepare for a meeting with new people. If you are thinking of changing jobs, that knowledge may even give you some guidance on whether that would be a "comfortable" area for you to explore. But do note that people of all types are found in all occupations. With that in mind, let's describe the different temperaments and make some suggestions for working with them.

NTs: Rationals

Many system administrators and systems designers are iNtuitive Thinkers (NTs). NTs are powerful visionaries, conceptualizers, and systems planners. They tend to be competent and consistent, firm minded, and fair. NTs generally are high achievers and are often non- conformists. They value knowledge, maintain objective perceptions, and are independent and intellectually curious. If you are an NT, you probably like to "word smith," you maintain high principles, enjoy complexity, and like to work independently. NTs are architects of change: they like to ask (and answer) "What would happen if?"

As leaders, NTs hunger for competency and knowledge, work well with ideas and concepts, are intrigued and challenged by riddles, see systematic relationships, and focus on possibilities through non personal analysis. NTs are responsive to new ideas.

As learners, NTs are interested in principles and logic, enjoy developing their own ideas, and find technology appealing. They tend to need constant success experiences, and exert constantly escalating standards on themselves and others.

Working with NTs

Work with these individuals by appealing to their intellect, love of technology, interest in principles and logic, and their desire for fairness. Recognize that to the NT, "Knowing" is of primary importance. Also know that "scratch an NT, find a scientist"[1]. You are well advised to provide your NT customers with the knowledge, information, and assistance they seek.

NFs: Idealists

People who are NFs are interested in meaning and significance, and in guiding others. As iNtuitives (a designation they share with many computer professionals), they see possibilities in relationships and institutions. Their F preference means that they are intricately concerned with others' feelings. Often referred to as "catalysts," NFs look for identity, growth, and for ways to make a better world (thus the designation idealists). As empathic individuals, they are fine people motivators and persuaders.

Working with NF's

When dealing with an NF in the user community, keep in mind that NFs are "becoming;" searching for self-realization of some higher good. NFs thrive on interactions with other people. When you deal with an NF, provide him or her with acceptance, caring, and support. If you give the NF "strokes," you will receive strokes in return. Your NF systems user would appreciate and respond to your praise, encouragement and recognition. NFs enjoy group interaction, prefer cooperation over competition, and like to focus more on people than the abstract. They learn best in personal, "face to face" situations. We believe that the more people-oriented you can make your orientation to NF users, the more the relationship will thrive. Avoid conflict, focus on the positive, be sympathetic and understanding, and help them avoid any situations in which they might feel guilty. 4.2.3. SPs: Artisans Persons with a preference for Sensing (S) and Perceiving (P) are known as Artisans. These individuals are concerned with variation and spontaneity. They seek action (doing), and often act as trouble shooters, problem solvers, and negotiators. SPs are practical and resourceful. They deal well with the immediate and are quick starters. They hunger for risk and excitement, desire freedom, and love to make a deal. In short, they love life and live it to the fullest (Remember Auntie Mame?).

Working with SPs

In their desire for variation, SPs may occasionally ignore protocol, procedures, regulations, and customs. If procedures are important to you or your work, you may need to reinforce this with SP users. Since SPs thrive on the verbal and the visual, and enjoy hands-on experiences, use these techniques to your advantage when working with them.

SJs: Guardians

SJs are often found in management positions and throughout the computer field. Guardians prefer to use their Sensing(S) and Judging (J) functions. They long for membership in meaningful institutions and the preservation of resources. SJs are traditionalists. They see their duty clearly, and are responsible and accountable. In fact, "serving" is important to them. SJs are precise, "take charge" people. They like and need organization, schedules, and the discipline of structure and authority.

Working with SJs

When working with SJs, use a very organized, structured, precise manner. Keep this in mind when you are explaining or teaching them about a system, software, or hardware. SJs like procedures; and so the more procedures you can build into the systems, the happier the SJ will be. SJs are sometimes thought of as somewhat rigid, so be aware that what appears to be rigidity is actually a strong level of comfort with organization and structure.

Looking at the "Whole" Person

As you learn to work with temperaments, remember that it is useful to look at individuals' other preferences as well. For example, if the person you are interacting with is gregarious and tends to initiate relationships, he or she may be an Extravert. If they wait for you to make the first move and then respond to that move, they may be Introverted. You may have to be more proactive with regard to initiating working relationships with Introverts than Extraverts. Other dimensions are their preference for structure and criteria used for making decisions. Generally, "role directive" people are those with a Thinking and/or Judging preference who are comfortable giving directives and structure to others. "Role informative" people may have a Feeling and/or Perceiving preference and may rely on others (you, for example, in your system administrator role) to make decisions.

Background on the MBTI

In this section we go into some of the work that has led to the current use of the MBTI, document some of its successes and discuss the views of some of its critics.

History

The MBTI has its roots in the work of C.G. Jung (1875 – 1961), a Swiss psychologist and psychiatrist. Jung believed that people are born with a predisposition to prefer certain functions over others. His research identified the attitudes of extraversion and introversion; he also identified the four psychological functions as Thinking, Feeling, Sensing, and Intuition.

Katherine Briggs, a non-psychologist, began observing and classifying differences among people around the turn of the century. When Jung's work was published in English in 1923, she was captivated by his theories and astounded at the similarities between his work and hers. Along with her daughter, Isabel Briggs Myers, Briggs began working to measure the differences between people. The MBTI was born as a result of their efforts to test Jung's theory and put it to practical use. The first indicator was developed in the early 1940s.

Briggs and Myers added two new dimensions to Jung's work. The dimension of perceiving and judging provided, in their view, a fuller understanding of Jung's work and a more complete comprehension of individual personalities.

Briggs and Myers spent the post-World War II years collecting additional data to support their theories and refine the MBTI. In 1956, the Educational Testing Service published the MBTI for use as a research instrument. New MBTI forms were developed over the next several years. Gradually, the instrument gained visibility; researchers and clinicians began to find it useful. In 1975, Consulting Psychologists Press, the current MBTI publisher, began publishing the indicator, and the instrument became more widely available. Today, it is one of the most widely used personality assessment tools in the United States.

Positive Perceptions

The MBTI has many positive factors, and is often the highlight of a training, team building, or career development program.

An Army career development program research study revealed that individuals who took the MBTI generally remembered taking the test, recalled their "letter" designations, and even remembered their scores. The respondents indicated that the results were true in that they confirmed what they already knew about themselves; and that the MBTI had either "very much" or "some" impact on their behavior. In this regard, they felt the MBTI made them more aware of themselves and others, which helped them adapt their behavior when relating to others. The instrument was so influential that respondents suggested additional follow-up as a means to improve the career program. [See 7, p. 96]

On an organizational level, Isachsen and Behrens, [8], describe their work with five different organizations. The MBTI helped each firm realize four significant accomplishments:

- They went from describing themselves as somewhat mediocre to a high- performance organization
- The MBTI legitimized the opportunity for all top and middle managers to influence the thinking of his or her peer group, as well as subordinates and superiors.

- Each firm became more focused and more strategic, as energy was freed up to bring about the conditions most favorable to their organization in the overall competitive environment.
- Each firm was better able to define the overall purpose of its organization, and a superordinate goal emerged for each firm.

Isachsen and Behrens generalize that the use of personality types applied to interpersonal relationships at work can help improve those relationships, reduce stress, and increase both teamwork and productivity.

Other authors draw similar conclusions. Otto Kroeger and Janet M. Thuesen, [see 5 and 6], advocate applying "Typewatching" to resolving conflicts and solving problems in the workplace. At the Federal National Mortgage Association (Fannie Mae), the MBTI was found to be a fundamental tool in team building and interpersonal skills work, [9].

Finally, the first author's work as a team trainer and facilitator has proven time and again that by understanding others' personalities, individuals can appreciate and capitalize on strengths and personality differences. This is the essence of team building.

Negative perceptions

As we've said before, type theory is not a panacea, and it does have its critics. The skepticism centers on three major issues: reliability, validity and effectiveness. [See 7, 10, and 11].

The reliability question is "Are the test results are repeatable?" If you take the test twice, will it give you the same four letter designation? Studies have shown that a significant number of people do change type over time. While this change may indicate "learning" to use a function that is not preferred, it does challenge the theoretical basis for the indicator. More problematic is the question of a person's ability to "fake" answers to come out with a predetermined result, perhaps the one that the hiring manager wants to see. Again, this has been shown to be possible and is an argument for warnings against using the MBTI as a selection instrument.

The validity question is "Does the test measure anything 'real'?" [10]. This is answered by looking at how well the MBTI correlates with other personality assessments. While we note that there are disagreements on this in the psychology literature, there is a large body of work that takes the MBTI to be a serious indicator of underlying personality traits. See [7], [2] and [11] for more details.

Finally, there's an effectiveness question, "Does the use of the MBTI in a training program change the behavior of the trainees?" There are no studies answering this question definitively, but there is a large population of practitioners committed to the use of the indicator, and there are organizations, e.g., [9], saying, in effect, "It works for us." The important answer to this question is whether it "works" for you.

Recommendations

Throughout this paper, we've given examples of using the MBTI as a descriptive tool, rather than a prescriptive one. By this we mean that you can use the MBTI dimensions to categorize and describe interpersonal differences, but that the MBTI in itself gives no hard and fast rules that take precedence over other factors in determining the outcome of interpersonal interactions. With knowledge of the MBTI differences, you can change your behavior, but standard warnings about the inability to directly affect others' behavior still apply. When used as part of a training program for a group of coworkers, the MBTI can provide a framework for discussing differences, resolving conflicts and capitalizing on strengths. When you are the only member of the group to describe differences using the MBTI dimensions, its power may be muted, but you can still reap some benefits from this knowledge. If you find yourself having the MBTI "imposed" on you, some knowledge of the strengths and weaknesses of the instrument can help you determine your reaction to the process and its outcome.

With all that in mind, we would like to make the following recommendations:

Avoid using the MBTI dimensions to justify or excuse communication problems. While the Myers-Briggs dimensions can provide insight into the source of conflicts, they do describe preferences, not absolute behavior patterns. Recognizing the preferences may give you clues as to what other patterns may be more effective. This information may be helpful in overcoming the communication problems, and lessen the need for continuing to live with the problem.

Be skeptical when using the MBTI dimensions for career counseling, job placement, applicant screening and selecting team members for complementary types. Given the reliability, validity and effectiveness issues noted above, you can be wary of attempts to pigeonhole people based primarily on the results of an MBTI assessment. At best, the MBTI should be used in conjunction with as much other information as possible on team members' skills and aptitudes.

Do use the MBTI to improve your self-knowledge. The MBTI has been shown to be an accessible and memorable way to categorize personality differences. Recognizing your own preferences, and how they may fit or not fit with your environment, can be a lens through which your own choices of work style can be clarified.

Try the MBTI for recognizing and working through differences with others, particularly if the MBTI is generally available to others in your workplace. When team members have knowledge of the others' preferences, discussions can be structured to take these preferences into account. When conflicts are framed as differences in Myers- Briggs preferences, often a solution to the conflict is apparent. When a team needs to develop a solution to a problem, consider implementation of Briggs Myers' "Z method":

1. Use **Sensing** to gather the facts of the case.
2. Use **iNtuition** to generate alternative interpretations.
3. Use **Thinking** to reach a tentative decision.
4. Use **Feeling** to validate the decision.

For more details on this method of problem solving, see [5].

Where to get more information

As noted before, the MBTI is a popular instrument in programs to build teamwork and to enhance communications. If you work for a corporation, you can check with your human resources office to see if they offer MBTI workshops. You may find public MBTI workshops offered through college and university psychology departments, community counseling centers, or private organizations. Expect a workshop explaining the four Myers-Briggs dimensions to take a half-day. A workshop introducing new skills based on the MBTI can take a day or more.

While the authors recommend a class or workshop that encourages the active participation that will reinforce new skills and insights, there are references that go into more detail than we've been able to do in this paper. See in particular, Kroeger and Thuesen, [5] and [6], and Keirsey and Bates, [1].

For more information on the MBTI itself, contact the Center for Applications of Psychological Type [CAPT] at 800-777-2278 or Consulting Psychologists Press at 800-624-1765.

Summary

This paper has focused on:

- An overview of the Myers-Briggs Type Indicator.
- An introduction to the 16 MBTI Types
- An explanation of Temperaments
- Hints for using the MBTI as a practical tool in your day to day work
- Some cautionary notes to avoid abuse or misuse of the MBTI, and
- Sources for additional information.

We encourage you to try using the MBTI to assist you in your day to day work. Our experience and research have convinced us that while the MBTI is not a panacea, it can be an invaluable problem-solving and communications tool to help system administrators do their jobs more effectively.

Acknowldgments

Myers-Briggs Type Indicator and MBTI are registered trademarks of Consulting Psychologists Press.

Type Table Format copyright Consulting Psychologists Press, Inc. Type descriptors copyright Otto Kroeger Associates.

References

[1] David Keirsey and Marilyn Bates, *Please Understand Me: Character and Temperament Types*, 5th edition, Gnosology Books Ltd., 1984.

[2] Avril Thorne and Harrison Gough, *Portraits of Type*, Consulting Psychologists Press, 1991.

[3] Center for Applications of Psychological Type, *CAPT-MBTI Atlas*, Gainesville, FL, 1986.

[4] Michael Doyle and David Straus, *How to Make Meetings Work*, Jove Books, 1976.

[5] Otto Kroeger and Janet M. Thuesen, *Type Talk at Work*, Delacorte Press, 1992.

[6] Otto Kroeger and Janet M. Thuesen, *Type Talk*, Dell Publishing, 1988.

[7] Daniel Druckman and Robert A. Bjork, eds., *In the Mind's Eye: Enhancing Human Performance*, National Academy Press, 1991.

[8] Olaf Isachsen and Linda V. Berens, *Working Together: A Personality Centered Approach to Management*, Networld Management Press, 1991.

[9] Catherine Taylor, "Home-Grown Team Building at Fannie Mae", *Employment Relations Today*, Autumn, 1992, v19n3, pp. 321-325.

[10] Ron Zemke, "Second Thoughts about the MBTI", *Training*, April, 1992, pp. 42-47.

[11] Isabel Briggs Myers and Mary H. McCaulley, *Manual: A guide to the development and use of the Myers-Briggs Type Indicator*, Consulting Psychologists Press, 1985.

Author Information

Reach Betty Jacob at PRC, Inc., 12001 Sunrise Valley Drive, Reston, VA 22091. Write to electronically at jacob_betty@prc.com.

Contact Nancy Shoemaker at PO Box 3562, Mississippi State, MS 39762. Her e-mail address is shoemaker@acm.org.

Upgrading 150 Workstations in a Single Sitting

Craig Manning – Computer Sciences Corporation, NASA/Ames Research Center
Tim Irvin – Industrial Light & Magic

ABSTRACT

As sites increase in size and complexity, the traditional methods used for operating system upgrades become too time-consuming and labor-intensive. Vendor-recommended upgrade procedures involve upgrading each machine from CDs, tapes, or (perish the thought) floppies, followed by an inevitable period of configuration. To upgrade a large number of machines efficiently, these methods must be abandoned in favor of a more automated cloning process. The upgrade procedure we developed was built around the *rdist* program, and allowed five members of the Workstation Group to upgrade 150 various Silicon Graphics workstations in 10 hours on a single Saturday. This paper will discuss the procedures developed, the planning and preparation involved, the upgrade itself, and the experience we obtained that will allow us to improve our method in the future.

Background

Traditional upgrade procedures are time consuming, and labor intensive. These procedures can be adequate for a small installation, but do not scale well into a larger environment. The vendors' method involves taking CDs, tapes, or floppies to each machine, booting the miniroot, partitioning the disk, making the filesystems, loading the operating system, installing the boot block, and finally performing local configurations. This process can take up to a couple of hours per machine. Upgrading a large number of machines requires a considerable amount of time and energy. This is an inefficient use of a skilled system administration staff, and an inconvenience to the user community. With vendors continually releasing new operating systems, this becomes increasingly more of a problem. Also, due to the manual nature of the upgrade process, the reliability of any particular installation is difficult to guarantee. Therefore, the more the process is automated, the quicker and more reliable it becomes.

Description of the site

The Numerical Aerodynamic Simulation (NAS) super-computing facility at NASA/Ames Research Center provides the scientific community with high speed processing (HSP) resources necessary to solve critical aeronautic problems. Besides the HSP computers, the NAS project currently maintains 200 Silicon Graphics (SGI) workstations, 100 Sun SPARCstations, and 6 Sun fileservers distributed over 13 subnets and 3 buildings.

The SGI workstations consist of 13 different hardware platforms. These disk-full machines are configured so that the / (root) and */usr* partitions contain only the operating system and are controlled solely by the system staff. A separate partition on each machine contains the user home directories. The third-party and local software, man pages, and demos are NFS mounted on the workstation from the fileservers.

Solution

Issues

In designing an automated upgrade process, we considered several issues. The process must minimize the downtime to the user community, while maximizing the reliability of the upgrade. It should allow us to clone a "live" machine (i.e., in multiuser mode) which enables us to do the upgrade from a central location, instead of having to visit each machine personally. The method should not only clone the new files, but also remove any old extraneous ones. It is crucial that the existing user environment remain intact, and that we can exclude certain files from being updated.

The two standard utilities, *dump/restore*, and *tar*, are effective at moving large amounts of data, but they do not adequately address these issues.

A *dump* piped to a *restore* would require that we start with clean filesystems on the target machine. This means that we would have to boot the miniroot, *mkfs* the / and */usr* filesystems, add entries for the source and target machines in */etc/hosts,* configure the ethernet interface, possibly add a default route, *mount* the / and */usr* partitions on a temporary mount point, and finally, execute

```
# rsh <source> dump 0f - / | \
             (cd /a; restore rf - )
```

and

```
# rsh <source> dump 0f - /usr | \
             (cd /b; restore rf - )
```

This method would have made it impossible to upgrade the machines from a central location, and it would have been nearly as time consuming as upgrading from vendor media.

The problems with using *tar* are a bit different. Since we would not necessarily have to start with a clean filesystem, the clone would not have to be done from the miniroot. But, we could not simply run

```
# tar cf - / |  rsh <target> \
         "cd /; tar xBf - "
```

from the source machine, since *tar* lacks the capability of dealing with open files. It simply copies over running binaries and shared libraries, which would cause the target machine to hang during the upgrade. Also, *tar* does not provide an easy method of excluding certain files from being replaced.

So, we turned to a third utility, *rdist. Rdist* is a program that

> . . . maintains copies of files on multiple hosts. It preserves the owner, group, mode, and modification time of the master copies, and can update programs that are executing. Rdist reads commands from *distfile* to direct the updating of files and/or directories[1].

Since this program can update programs that are executing, we would not have to worry about the target machine hanging during the upgrade. This is possible because *rdist* will copy a new file onto the machine using a temporary name, and then will use *rename*(2) to replace the old file with this new one. Since the disk space used by these replaced, but still open, files is not reclaimed until after the files are closed, the system must have enough free disk space to accommodate the extra overhead. To reduce the amount of extra disk space needed, we minimized the number of open files by rebooting the machine, and killing off all unnecessary daemons. We also did some disk cleanup on some of our machines to increase the available disk space.

The *-R* option to *rdist* instructs it to remove "extraneous files." This provides an easy method of removing files not found on the source machine. The *distfile* primitive *except* allows us to specify files and directories that should be excluded from the clone. This lets us preserve machine dependent files, such as */etc/passwd*, */etc/hostname*, and */etc/fstab*, as well as the user home directories.

One problem with *rdist* that we needed to solve is its inability to copy device files. To work around this we used a *special* primitive in the *distfile* to instruct *rdist* to run */dev/MAKEDEV* when the upgrade finished. As an additional safety precaution, we also used the *special* primitive to rebuild the kernel from the just updated kernel source and object files.

Procedure

The scripts that were developed enabled us to log onto a machine that both the target and source machines trust, such as a fileserver, and run

```
# upgrade <source> <target>
```

Approximately 45 minutes after starting *upgrade*, the target machine will reboot with the new operating system running. The various scripts, and the *distfile* that we used appear in the Appendices. Behind the scenes of *upgrade*, the following steps take place:

1. *Fixdest* is run on the target machine. It places the source machine's name in */.rhosts* on the target machine, and kills unnecessary daemons.
2. *Doit* is run on the source machine. It copies a *shar* file that contains the script *run.dist.upgrade* and the *distfile dist.upgrade* from the fileserver. The files are extracted from the *shar* file. *Doit* then runs *run.dist.upgrade*, which:
   a. compares the architecture of the source and target machine to ensure they are identical;
   b. unmounts unneeded filesystems on the target machine;
   c. copies the *rdist* binary to the target machine, to ensure that the versions match;
   d. starts the *rdist* on the source machine, which does the upgrade using the *distfile dist.upgrade*, and reboots the target when finished.
3. Once the target reboots, *upgrade* calls *cleanup* to do final cleanup and sanity checking, on the target machine.

Planning

Since each different SGI platform requires a separate OS build, our scripts had to be tested on each of them. Much of the testing went on over a period of a couple of months. The scripts would be tested on new machines as they arrived, or on machines whose user would request an early upgrade. The timing of this worked out well, since during this period we were still evaluating the new release, and we were dealing with some open bug issues with the vendor.

We started the planning process about a month before Upgrade Day. The first key issue was to upgrade and thoroughly test a "master" machine for each platform. Whenever possible, we used the system administrators' machines for this stage. This allowed us to spot any potential problems immediately, and to reduce the inconvenience to our users.

After determining the number of people we would have available that day, we set about creating a schedule for the upgrade itself. Many factors were critical to the schedule: the platform types to upgrade on each subnet, the load on the "master" machines, the subnets of the "master" and target machines, and the available bandwidth of each subnet. The schedule was broken into phases, with each person getting a list of approximately five machines to upgrade per phase. This list showed each machine and its corresponding "master." Each phase was to be completed before the next one started. This would allow machines upgraded during one phase to become "master" in later phases. We wanted to limit cross-subnet traffic as much as possible, so the schedule was set up to build "master" machines on each subnet during the initial phases. The final plan had each of the five staff members upgrading thirty machines over six phases.

We notified the user community a couple of weeks ahead of time, to allow them to schedule around the weekend. We scheduled down time on all machines from Saturday morning until Monday morning, to give us plenty of disaster recovery time, should we need it.

The week before the upgrade, *at* jobs were set up on all machines to reboot them at 6:00 am Saturday morning, and install an */etc/nologin* to prevent users from logging in during the upgrade.

A few of the older machines were tight on disk space, but we were able to move some programs onto the fileserver. We tried to have a 10MB cushion in */usr* on each of the machines. We also made several checks on the "master" machines, to make sure nothing had changed after their initial upgrade.

Upgrade Day

On the fateful day, the machines rebooted as planned at 6:00 am. The staff assembled at 9:00 am to get their schedules and instructions. Once we determined that everything was ready, the signal was given for everyone to start the first phase. Once this initial phase was completed we did some rudimentary sanity checks on the upgraded machines, and corrected a few minor bugs.

We then continued with the remaining phases. During the process we monitored the bandwidth utilization on one of our key subnets, so that we would get an idea of the impact it was having. Each machine being upgraded was using approximately 10% of the bandwidth. While five machines per subnet was pushing the network, it was handling this level of traffic without much difficulty.

At the beginning, when we had only a few "master" machines of each platform type, the process was a bit slow. The load average of these machines rose while trying to clone a few machines at once. This problem was alleviated quickly as the number of "master" machines increased.

During the upgrade, we discovered a few machines that had inadequate free disk space. This would become quickly evident, as *rdist* began complaining, "No space left on device." The problem on these machines was generally caused by additional packages that had been installed, or large crashdump files in */usr*. On these machines, we would free up some space on the partitions, and restart the upgrade — which would complete very quickly, since the bulk of the files had already been installed.

Final outcome

At the end of the day, we performed some sanity checking on each machine. This included running *uptime* to make sure the machine had rebooted, running *uname* to ensure the machine had been upgraded, and checking that *xdm* was running to be sure that users could login. We double checked to make sure everything had been accomplished, and that there had been no major problems. The entire process took the five of us 10 hours, and our network bandwidth on each subnet hovered around 40-50%, with a couple of short-lived 70% spikes. We were extremely satisfied with our results. After upgrading 150 machines, only four experienced problems during reboot.

On Monday, most users experienced no problems whatsoever. There were a few machines that were missing a few files. After going through the logs, we determined that each of these was the result of a machine that had run out of disk space during the latter stages of the upgrade. The person upgrading this machine would miss this fact, and the machine would then become a "master." An incomplete upgrade would then be propagated to other machines. Once we saw what happened, we could easily track down the affected machines, using the log files and schedule, and quickly repair the damage. We had fixed all the problems by noon.

What we learned

We are very pleased with the process we used, and would only make minor changes next time.

We would make sure that all the destination machines have sufficient free disk space before starting. It should be possible to automate this process by removing unknown trees, and cleaning out crashdumps. This would prevent the cases where the disk space filled up and the *rdist* had to be run again.

Also, we would reduce the amount of output saved in the log files, and develop a few tools that would parse it into an easy-to-read format. This would help prevent incomplete builds from passing unnoticed, and would keep these machines out of the "master" machine pool until ready.

Lastly, we would prepare a few more "master" machines ahead of time. This would help spread the load across a larger set of machines for the initial rounds.

Conclusions

Comparing this mass upgrade to those of the past, we have concluded that automating with *rdist* is an efficient and reliable method of upgrading a large number of machines. Since multiple machines could be upgraded simultaneously, the upgrade time was significantly reduced. We were able to complete in one day what formerly required several months to accomplish.

This method required a good bit of initial planning, and script building, but this work was finished before any of our users were affected. The inconvenience to our users was kept to a minimum, and most of them completely forgot that their machines had been upgraded over the weekend – a tremendous compliment.

We would definitely use this approach again. The ease with which each machine was upgraded, the number of machines that we were able to upgrade in one sitting, the speed at which it was done, and the reliability of the final product convinced us that automating workstation upgrades using *rdist* is a valuable and successful method.

Acknowledgments

We thank David Clark and Josh Goldenhar for their initial work in building some of the scripts. We also thank our co-workers in the NAS Workstation Group, Robert Brophy, Louise Kokinakis, Michael ‘‘Seib’’ Seibl, and Arnold Yee for their invaluable assistance in performing this upgrade. Thanks also to Karen Castagnera, Jean Clucas, and Ken Beyer for reviewing this paper.

Reference

[1] Computer Systems Research Group, ‘‘rdist(1),’’ Unix User’s Reference Manual, University of California, Berkeley, 1986.

Author Information

Craig Manning, a San Francisco Bay Area native, received a BS in Computer Sciences from California State University, Hayward in 1988. He worked for two years at SRI International as a Sun Systems Administrator. In 1990, he went to work for Computer Sciences Corporation at NASA/Ames Research Center for the NAS project as a Systems Administrator for Suns and SGIs. He was promoted to Lead Systems Administrator for the Workstation Group in 1991. He is now working in the Workstation Development Group as a Systems Programmer. He can be reached by electronic mail at manning@nas.nasa.gov.

Tim Irvin received a BS in Computer Science from The University of Tennessee, Knoxville in 1990. He spent a very cold year and a half in Hanover, NH, as a Systems Administrator for Project NORTHSTAR at Dartmouth College. He moved to sunny California in 1992, to work for Computer Sciences Corporation at NASA/Ames Research Center as a Systems Administrator in the NAS Workstation Group. He is now the Senior Systems/Network Administrator at Industrial Light and Magic, in Marin County, CA. He can be reached by e-mail at tirvin@netcom.com.

Appendix A: *upgrade*

```
#!/bin/sh

NEWOS=4.0.5

cd /u/wk/irvin/Upgrade

source=’rsh $1 hostname’
dest=’rsh $2 hostname’

if [ -z "$source" ]
then
   echo "$1 unknown machine"
   exit 1
fi

if [ -z "$dest" ]
then
   echo "$2 unknown machine"
   exit 1
fi

sourcevers=’rsh $source uname -r’
destvers=’rsh $dest uname -r’
```

```
if echo $sourcevers | grep $NEWOS > /dev/null
then
   :
else
   echo "$source is not at $NEWOS.  It is running $sourcevers."
   exit 1
fi

if echo $destvers | grep $NEWOS > /dev/null
then
   echo "The destination machine $dest is running $destvers. "
   echo "To cancel type <CTRL-C>."
   echo -n "Press enter to continue with the upgrade: "
   read foo
fi

hostname=‘hostname‘
echo "Upgrading $dest from $source.  $dest is running $destvers.  $source is"
echo "running $sourcevers.  If this is not correct type <CTRL-C> now."
echo -n "Otherwise, press enter to do it: "
read foo

/usr/ucb/rcp ./fixdest $dest:/tmp/fixdest
rsh $dest /tmp/fixdest $source

/usr/ucb/rcp ./doit $source:/tmp/doit
rsh $source /tmp/doit $dest

echo "$dest has been upgraded.  It should now be rebooting.  Hit enter when"
echo "the reboot is finished."
echo -n "> "
read foo

/usr/ucb/rcp ./cleanup $dest:/tmp/cleanup
rsh $dest /tmp/cleanup

echo " "
echo "df listing for ${source}:"
rsh $source "/bin/df -k / ; /bin/df -k /usr | tail -1"

echo " "
echo "Compare the above df's.  The / and /usr partitions from the 2 machines"
echo "should be approximately the same size."
echo " "
echo "======== The End ========="
```

Appendix B: *fixdest*

```
#!/bin/sh

killit () {
   echo "Killing $1"
   ps -ef | awk ’{print $2 "/" $8}’ | awk -F/ ’{print $1 ":" $NF}’ | \
          awk -F: ’$NF == "’$1’"{print $1}’ | xargs kill
}

echo "Fixing rhosts on DESTINATION"
echo "${1}.nas.nasa.gov root" >> /.rhosts

DAEMONS="sendmail lpd cron xntpd cdromd nqsdaemon"
for i in $DAEMONS
do
  killit $i
done
```

Appendix C: *doit*

```
#!/bin/sh

cd /u/wk/nasops
mkdir $1
cd $1
/bin/rm -fr rdist
rcp donald:/u/wk/dist/upgrade.shar .
sh upgrade.shar
cd rdist
./run.dist.upgrade $1
cd /u/wk/nasops
/bin/rm -fr $1
```

Appendix D: *run.dist.upgrade*

```
#!/bin/sh
# This bourne shell script is the master distribution script on
# SOURCE systems for upgrades.  This script calls:
#        dist.upgrade    (rdist script)
#
# USAGE:         run.dist.upgrade TARGET [-v]
#
#
trap "echo $0: exit on intr; exit" 1 2 15

#........................SCRIPT-WIDE VARIABLES...................

RDIST=/usr/ucb/rdist  #always use the same rdist
PROG=‘basename $0’    #calling prog
SOURCE=‘hostname’     #rdist source
TARGET=""             #rdist target
RESP=""               #used by getresp for interactive
T=0; F=1;             #True and False
SCRIPT=script.ug      #used for script logging
SOLOG=""              #upgrade source log
MODE=""               #verify only or rdist

#......................................................................
# banner
#......................................................................
ban()
{
echo
echo "===== =====> $1   "
}

#......................................................................
# usage
#......................................................................
usage()
{
          #0 args: prompt for options
          #1 args: $1 is system name
          #2 args: $1 as above, $2 is -v

echo "Usage: $PROG systemname;  $* "
}

#......................................................................
# for interactive
#......................................................................
```

```
getresp()
{        #yes==0, no==1
echo "$1 y|n ?  \c"

read resp
case $resp in
      y|Y) RESP=$T
        break;;
      n|N) RESP=$F
        break;;
        *) echo invalid response
           echo "$1 y|n ? \c "
           read resp
           case $resp in
              y|Y)  RESP=$T
                    break;;
              n|N)  RESP=$F
                    break;;
                *)  echo
                    echo
                    echo
                    echo "2nd try, invalid response "
                    echo "Sorry, exiting"
                    exit 9
                    break;;
           esac
esac
}
#......................................................................
# if host not in hosts || hosts not reachable then exit
#......................................................................
vertarget()
{
TARGET=$1

grep $TARGET /etc/hosts >> /dev/null
if [ $? = 1 ]
then
        echo "‘basename $0’: Sorry, $TARGET not found in host table"
        exit
fi

echo "Checking network connection to $TARGET"
rsh $TARGET date >> /dev/null
if [ $? = 1 ]
then
        exit                     #Connection problems
fi
}
#......................................................................
# This ensures that the same version of rdist is being used !
#......................................................................
mvrdist()
{
ban "Moving rdist binary"
rsh $TARGET mkdir /usr/bsd
rsh $TARGET ln -s /usr/bsd /usr/ucb
rcp $RDIST $TARGET:$RDIST
}
```

```
#.....................................................................
# Check the source system
#.....................................................................
chksource()
{
# .............................................# same hardware type ?

TARGU=‘rsh $TARGET uname -m’
SORU=‘uname -m’

ban "Check the cpu types match"
if test $TARGU = $SORU
then
        echo $SORU and $TARGU: identical architectures
else
        echo
        echo "Local host has $SORU while target has $TARGU cpu "
        echo "CPU Architectures do not match"
        exit
fi
# .............................................# stray kernels ?
if [ "$INTERACT" != "N" ]
then
        ban "Check for extraneous kernels in /"
        ls -l /unix*
        getresp "\t Quit $PROG to remove extraneous kernels?"
        if [ $RESP -eq $T ]
        then
            echo exiting script
            exit 1
        fi
fi
# .............................................# make dir for log file
if [ ! -d /usr/spool/log/install ]
then
        mkdir /usr/spool/log/install
fi

SOLOG=/usr/spool/log/install/RDIST.SOURCE.SYSTEM.LOG.$TARGET
touch  $SOLOG
# .............................................# check other mount points
if [ "$INTERACT" != "N" ]
then
       ban "Check for strange mount points on SOURCE machine:"
       /etc/mount | grep -v wk
       getresp "\t Quit $PROG to unmount non-standard file systems?"
       if [ $RESP -eq $T ]
       then
          echo exiting script
          exit 1
       fi
fi
}
#.....................................................................
# look for previous rdist attempts, log directory
#.....................................................................
checktarget()
{
ban "Rm left-over checkpoints in $TARGET:/usr/tmp "
rsh $TARGET ls -la /usr/tmp/rdu.*
rsh $TARGET /bin/rm -f /usr/tmp/rdu.*
rsh $TARGET "mkdir /usr/spool/log /usr/spool/log/install > /dev/null "
ban "Adding .rhosts entry"     #This will be copied to target
echo "$SOURCE root" >> /.rhosts
echo "$SOURCE.nas.nasa.gov" >> /.rhosts
echo "umounting /u/wk"
rsh $TARGET "/etc/fuser -k /dev/wk"   > /dev/null
rsh $TARGET "/etc/umount -av"         > /dev/null
rsh $TARGET "/etc/umount /dev/wk"     > /dev/null
# Remove directories and files that are to be linked to the fileserver
# or other files.
rcp resolv.conf.all ${TARGET}:/usr/etc
rcp rm.links link_list ${TARGET}:/
rsh $TARGET 'cd /;/rm.links'
}
#.....................................................................
# rdist in verify mode
#.....................................................................
distv()
{
ban "$PROG: A script named $SCRIPT.ver.err.$TARGET will be started"
sleep 1
if [ -f $SCRIPT.ver.err.$TARGET ]
then
   cat $SCRIPT.ver.err.$TARGET >> $SCRIPT.ver.err.$TARGET.o > /dev/null
   /bin/rm -f $SCRIPT.ver.err.$TARGET     > /dev/null
fi
if [ -f $SCRIPT.verifiy.$TARGET ]
then
   cat $SCRIPT.verify.$TARGET >> $SCRIPT.verify.$TARGET.o  > /dev/null
   /bin/rm -f $SCRIPT.verify.$TARGET  > /dev/null
fi
$RDIST -f dist.upgrade -v -d TARGET=$TARGET | \
    tee $SCRIPT.verify.$TARGET 2>$SCRIPT.ver.err.$TARGET
echo "verification complete \n"
}
#.....................................................................
# rdist: this is the real thing
#.....................................................................
dist()
{
ban "$PROG: A script named $SCRIPT.inst.$TARGET will be started"
sleep 1
if [ "$INTERACT" != "N" ]
then
   getresp "Ready for the real thing"
   [ $RESP -ne $T ] && exit
fi
sleep 3
$RDIST -f dist.upgrade -d TARGET=$TARGET | \
    tee $SCRIPT.install.$TARGET 2>$SCRIPT.inst.err.$TARGET
}
```

```
cleanup()
{
echo "$TARGET has now been upgraded by ‘hostname‘" >> $SOLOG.
}
#......................................................................
# getinfo
#......................................................................
getinfo()
{
echo "$PROG:    ----- Interactive Mode -----"
ban "Enter Workstation Name:     \c:"
read TARGET
vertarget $TARGET
getresp  "\t Use verification mode?"
if [ $RESP -eq $T ]
then
   MODE="V"
   echo "$PROG will be run in verification mode"
else
   MODE="R"
   echo "$PROG is in FULL DISTRIBUTION MODE"
   echo "DATA will be overwritten on $TARGET"
fi
}
#......................................................................
# main      Main calling area
#......................................................................
#.... determine if running in verify mode
#.... check system for "clean flag" for log files
case $# in
   0) getinfo   #prompt for all options
      break;;
   1) vertarget $1    # first arg is the system name
      MODE="R"
      INTERACT="N"
      break;;
   2) case $2 in       # running in verification mode
        -v) vertarget $1
            MODE="V"
            break;;
         *) usage "$2 invalid"
            break;;
      esac
      break;;
   *) usage invalid arguments
      break;;
esac
case $MODE in
   V) chksource
      distv
      break;;
   R) chksource
      checktarget
      mvrdist
      dist
      cleanup
      break;;
```

```
    *) usage "$*: Misplaced argument"
       exit;;
esac
```

Appendix E: *dist.upgrade*

```
SOURCE = ( / )

#        >> Standard root exceptions
X_ROOT = ( ${SOURCE}/{debug,dev,tmp,u,lost+found}
           ${SOURCE}/etc/{aliases,config/glb,config/llb,config/maker,
                  config/netls,config/netlsd.options,cshrc,exports,
                  fstab,group,hosts,hosts.equiv,hosts.lpd,init.d/local,
                  init.d/maker,init.d/mathserver,init.d/matlab,
                  init.d/nck,init.d/netls,init.d/wavefront,lmboot,
                  localhosts,localnetworks,motd,mtab,networks,nmap,
                  passwd,passwd.sgi,printcap,rc0.d/K10maker,
                  rc0.d/K11mathserver,rc0.d/K12matlab,rc0.d/K13netls,
                  rc0.d/K14nck,rc0.d/K15wavefront,rc2.d/S51maker,
                  rc2.d/S52mathserver,rc2.d/S53matlab,rc2.d/S54nck,
                  rc2.d/S55netls,rc2.d/S56wavefront,rmtab,
                  sys_id,ttytype,lvtab,utmp,wtmp,xutmp} )

#        >> Standard usr exceptions
X_USR = ( ${SOURCE}/usr/{adm,mail,netls,spool,u,tmp,lost+found}
          ${SOURCE}/usr/etc/{gated.conf,glbd,llbd,lb_admin,ncs,netlsd}
          ${SOURCE}/usr/lib/{aliases,libnck.a} )

files:
${SOURCE} -> ${TARGET}
          install -R  / ;
          except ( ${X_ROOT} ${X_USR} );

DEV   = ( /dev/MAKEDEV )

dev:
${DEV} -> ${TARGET}
         install  /dev/MAKEDEV ;
         special "cd /dev;/dev/MAKEDEV ";
         special "chown root.root /dev/kmem /dev/mem /dev/mmem";
         special "chown bin.bin /usr";
         special "cd /; lboot -u /unix.install";
         special "touch /unix";
         special "chmod 755 /unix";
         special "/etc/shutdown -y -g0 -i6";

# =================== end of rdist file
```

Appendix F: *cleanup*

```
#!/bin/sh

echo "Cleaning up DESTINATION machine.  Ignore errors..."
/bin/rm -f /core
/bin/rm -rf /lost+found/* /usr/lost+found/*
/bin/rm -f /tmp/cleanup
/bin/rm -f /tmp/doit
/bin/rm -f /rm.links
/bin/rm -f /link_list
echo " "
echo " "
echo "df listing for " ‘hostname‘ ":"
/bin/df -k /
/bin/df -k /usr | tail -1
```

Real-World Gigabit Networking

Jeff Pack – Grumman Data Systems

ABSTRACT

In the past few years, high-speed networking has become a reality with such technologies as HIPPI, UltraNet, and FDDI. The Fleet Numerical Oceanography Center (FNOC) in Monterey, CA has recently implemented a high-speed LAN with these technologies. This paper presents a "real-world" review of FNOC's experiences with these technologies and their efforts to utilize the network in a mission-critical application – environmental products and forecasts for the U. S. Navy. The hosts include two Cray Research (CRI) supercomputers, a high-speed frame buffer, and several Sun servers that act as gateways and peripheral servers.

Performance testing and tuning of the hosts and network is emphasized, along with examples of utilities and applications optimized for the high-speed network. The effects of host configuration are reviewed, along with future plans for improvement.

Introduction

.The Fleet Numerical Oceanography Center (FNOC) is the U. S. Navy's primary numerical prediction center for automated oceanographic, atmospheric, and applied products. FNOC is one of a half-dozen centers worldwide running global and regional atmospheric models on an operational basis, and is the world leader in performing oceanographic and coupled air-ocean modeling operationally. Being an operational center means no downtime is acceptable; redundancy and reliability are designed into each component of the system.

High-speed communications among the various computers is vital for the success of the center. Due to the distributed design of the system, large amounts of data are moved between hosts during the modeling run. HIPPI connections between the Cray mainframes and UltraNet connections to the Sun servers provide the bandwidth required for a successful result.

Background

The current production system at FNOC is anchored with a CDC Cyber 205 that has been in use since 1982. In order to improve and enhance the capabilities of FNOC, a program was started in 1990 to install new large scale computing resources and associated support hardware and software. Grumman Data Systems was awarded the contract to provide and integrate the new program. A Cray Research, Inc. Y-MP 2E was installed in November 1991 to operate the Empress database management system (DBMS) for the larger compute server, the Cray C90, which was installed in September 1992. Providing network and communications access, software maintenance functions, and graphics support are several Sun workstations. Current plans call for concurrent operations of the old and new systems beginning in January 1994 with completion of the switch to the new system by June 1994.

The high-speed LAN consists of the two Cray mainframes, three Sun servers, a Cisco router, and a frame buffer. The Cray mainframes have 2 high-speed network HIPPI channels. One of the HIPPI paths is a direct link between the Cray machines and the other HIPPI path goes to an UltraNet 1000 hub, which also links the Sun servers and the frame buffer. The default path between the Crays is the direct HIPPI connection, but the Ultra HIPPI link is always active for redundancy or special applications. An UltraNet/Cisco interface provides connectivity to other LANs.

Discussion

HIPPI

The high performance parallel interface (HIPPI) was first developed at Los Alamos National Laboratory and has since evolved into an ANSI standard. The physical characteristics of HIPPI were once described by Dr. David Clark of MIT as "an 800 Mbit/sec RS-232"[1]. HIPPI is therefore not a conventional LAN but an interface that, when coupled with a switch, can be used as a high speed LAN. HIPPI also provides a "double-wide" capability that doubles throughput to 200 MB/sec by simply doubling the cables used from 2 to 4.

The direct HIPPI link between the two Crays at FNOC is a double-wide connection that provides 1.6 Gb/sec throughput. Using TCP and **nettest,** a CRI-enhanced network testing tool (similar to **ttcp),** we have achieved throughput of 120 MB/sec, or 0.96 Gb/sec. With creative interpretation of significant figures, we have achieved a "gigabit" data transfer.

[1] Interop Tutorial, *The Art and Engineering of Protocol Performance*, August 1993

Sustained throughput numbers are around 90 MB/sec. Compare this to the comment that Greg Chesson made at USENIX in 1987 suggesting that host software could never get TCP to run at 10 Mb/sec[2].

Evidently, TCP has evolved along with network hardware. One example of this evolution is the adoption of the Van Jacobson "slow-start" TCP window algorithm in the late 1980s to provide congestion control. The impact on the NSFNet was quite significant. Cray Research has done a lot of work on their implementation of TCP in order to achieve gigabit rates, including increasing the sequence space and window size, vectorizing the checksum, and reducing the number of memory cycles on the data, to the point where memory cycle speeds are becoming a limiting factor. Work continues in the TCP research areas of congestion control and traffic shaping.

Another factor in overall performance is the interfaces between applications, the DBMS, and the network protocol. At FNOC, the most important interface is the DBMS. If the DBMS cannot provide data to the model over the HIPPI channel in a timely manner, the model has to wait and performance of the entire system is degraded. To achieve the performance required, the Empress developers have rewritten and tuned sections of the DBMS to Cray's hardware, TCP implementation, and the characteristics of the HIPPI channel. For example, Empress originally used bit masks of one byte to set decision flags, but since the Cray architecture operates best with 64-bit entities, the one-byte masks were exchanged for 64-bit masks. They also increased the maximum packet size to correspond with the 65496 MTU size of Cray HIPPI.

The areas that we have concentrated testing in are the "well-known" applications of **ftp** and NFS. The **ftp** program is widely used at FNOC to transfer data between hosts. Due to the large amount of data being processed, any optimization of **ftp** is welcome. Cray has added some extensions that, at least between two Cray machines, provide very high performance. Some of these extensions include preallocation of disk space for incoming data, user-defined buffer sizing and management, and user-defined TCP window sizing. Using the default settings, here at FNOC the typical transfer rate during the day shift averages 14-16 MB/sec. By tweaking these various parameters, we can essentially transfer data at disk channel rates, currently 20 MB/sec on FNOC's single-channeled disks, as shown in Table 1. FNOC does not have disk arrays, but other Cray sites with disk arrays have seen rates in the 60-80 MB/sec range.

Cray HIPPI FTP Rates	
Default	Optimal
14 MB/sec	20 MB/sec

Table 1: Cray HIPPI **ftp** Rate

NFS is another story. As you're probably aware, NFS is not a performance-tuned protocol. Synchronous writes and an 8 KB data block do not add up to high performance. Using the default parameters, NFS writes at approximately 190 KB/sec and NFS reads run at around 1 MB/sec. For NFS mounts between Cray hosts, the block size can be increased to 32 KB. When the larger block size is coupled with file system cache, the rates as shown in Table 2 increase to 750 KB/sec for NFS writes and 6.3 MB/sec for NFS reads. Due to these uninspiring numbers, NFS is currently not used for any operations during the modeling run, but is used for convenient data transfers that are not time-dependent.

Cray HIPPI NFS Rates		
	Default	Optimal
Read	1000 KB/sec	6300 KB/sec
Write	190 KB/sec	750 KB/sec

Table 2: Cray HIPPI NFS Rates

When working with HIPPI, maximize the packet or block size. Best performance with disk I/O on Cray machines is achieved when the block size is a multiple of the disk sector size. If asynchronous I/O is a feasible option for your application, it should be investigated. Proper use of I/O buffers can improve performance dramatically. TCP window size can also be a factor. Vendors are improving their TCP and NFS implementations, so it pays to check with them.

UltraNet

Ultra Network Technologies was founded in the mid-1980s and provides a LAN that specializes in using external high-performance adapters and high-speed data links that connect to a central hub. The advantages that UltraNet provides include:

- Offloading of network functions to external adapters;
- An OSI-based transport protocol that uses large packet sizes; and
- A sockets compatibility library that allows TCP/IP socket applications to utilize the high-speed transport protocol.

The UltraNet configuration at FNOC includes UltraNet HIPPI connections to each Cray, an UltraNet frame buffer and UltraNet Link connections to three Sun servers and a Cisco router. The HIPPI connections and the frame buffer can operate at 100 MB/sec, the Sun links can operate at 31.25 MB/sec and the Cisco router link can operate at 15.6 MB/sec.

[2] Interop Tutorial, *Gigabit Network Architectures*, August 1993

The heart of the UltraNet system is the UltraNet 1000 hub, which has a 1 gigabit backplane that interconnects all of the adapters in the hub. Redundant power supplies and hot-swappable boards provide good uptime capabilities. The hub houses host adapters (HIPPI), serial link adapters to interconnect hosts that have an on-board host adapter (Suns and Cisco), and network device adapters (frame buffer). Each hub adapter has a separate ''Protocol Processor'' which processes the packets over the backplane.

In testing performance of the UltraNet, we used the same basic ''real-world'' applications of **ftp** and NFS. Performance numbers for **ftp** transfers between the Cray hosts using the UltraNet HIPPI interface are similar to the direct HIPPI interface. Small file transfers are actually a bit faster over the UltraNet due to the smaller MTU size (33280). NFS shows similar results, with the rates being nearly identical.

The UltraNet frame buffer is technically a network device, since it is installed in the UltraNet 1000 hub. FNOC has applications that utilize the frame buffer and can drive it close to 100 MB/sec. This is probably the only practical application that runs at that speed, though.

The area in which UltraNet shows dramatic improvement over normal LAN technologies is in the Sun **ftp** tests. The Sun UltraNet host adapter at FNOC is a VME-bus card that installs in the server's card cage. Fiber-optic cable connects the card to a link on the UltraNet 1000 hub. The average **ftp** transfer over IEEE Ethernet on a quiet network between two Sun SPARCStation 10s is around 1 MB/sec. The theoretical peak of Ethernet is around 1.3 MB/sec. Using native (OSI) Ultra on a Sun 4/470 with IPI disks, the **ftp** rate is 3.5 MB/sec, which is about the speed of the disk channel. Using the Ultra with TCP/IP slows the rates to 2.8 MB/sec, still respectable (see Table 3). Ultra has announced an S-Bus card which we plan to beta test. FNOC does not have a faster Sun with differential SCSI disks connected to the UltraNet, but a faster host and disks coupled with the new S-Bus card should increase this rate further.

Sun FTP Rates	
Media/Protocol	Rate
Ethernet	1 MB/sec
Ultra TCP/IP	2.8 MB/sec
Ultra Native	3.5 MB/sec

Table 3: Sun **ftp** Rates

Of course, there are tradeoffs with the UltraNet LAN. The highest speeds are achieved with the Ultra proprietary protocol and, although all TCP/IP applications work under Ultra, they are a bit slower than their UltraNet ''native'' cousins. The company itself is under change, having been purchased by Computer Network Technology (CNT) so support could become an issue. But, if your primary motive is performance and/or connectivity to HIPPI-based mainframes, UltraNet is certainly worth a look.

UNIX and High-Speed Networks

System administrators often are also network administrators to some degree, since the network directly affects the performance of hosts. When the performance on a system is bad, the users call the system administrator. Here are some system administration details that will come into play when dealing with high-speed networks, whether it be HIPPI, UltraNet or FDDI.

Think BIG. If your data can be kept in large chunks through the network, it will have a higher overall throughput. Consider the analogy of putc() vs write(fd, buf, SOME_BIG_NUMBER). System buffers should be checked to verify that adequate space is available. On most implementations of UNIX, things like mbufs and streams are allocated dynamically, but the maximum amount of space available for system buffers is typically defined in the kernel. If your applications are mostly TCP-based, the TCP buffer sizes should be checked and increased if necessary. Ask your particular vendor about recommended tuning methods (your mileage may vary). Ask about the optimum MTU size for your particular network media. Often, it may be better to operate at a smaller MTU size than the maximum, due to system configuration or IP packet fragmentation.

If you have the luxury of having a system to test with and time to do it, it can pay big dividends. In the case of FNOC, by testing and tuning the Empress DBMS, the performance has increased by an order of magnitude or better.

Most system administrators don't have a teaching degree, but they end up doing a lot of end-user education; and utilizing a high-speed LAN is no different. A knowledgeable user base results in effective use of the resources.

Once you get a set of performance numbers for typical applications, check with similar sites or your vendors to see if your numbers correspond. Sometimes faulty hardware doesn't break and stop, it just slowly degrades. We were able to compare data rates with other supercomputer sites verifying our results.

Future Plans

FNOC has had fairly good success implementing the high-speed LAN and optimizing our applications for it. There are plans to install an FDDI backbone for the site-wide LAN and another FDDI backbone, which will be coupled with the high-speed LAN via the Cisco FDDI/UltraNet gateway. Cray has plans to enhance the HIPPI device driver for the next release and decrease the overhead processing on the direct HIPPI connection. We are watching the ATM and FiberChannel technologies for progress, as well as considering the benefits of a disk array for the Cray C90. Redundancy and reliability are very important, as well as performance.

Conclusions

The word ''gigabit'' should be used with care around today's technology. Although it is possible to achieve the equivalent of 100 Ethernets on one wire (or fiber), the only ''real world'' application that FNOC uses that rate for is graphics display to a frame buffer. Speaking as a UNIX system administrator, however, the ability to perform an **ftp** at disk I/O rates is nice. NFS should be considered a ''read-only'' option and only as a convenience.

The operation of TCP/IP on high-speed LANs requires an extension and re-implementation to the TCP code and, in some cases, extending and rewriting the applications. An example of this software evolution is the Cray extensions to **ftp** for defining window and buffer sizes. UltraNet shows that offloading network processing from the CPU to specialized hardware is another option to consider.

To maximize your investment in high-speed LANs you'll have to pop the hood and get your hands dirty. Tuning the operating system and applications is a given. High quality vendor support is mandatory and expected. Educating the users about the best ways to use the high-speed LAN typically falls on the system administrator.

The future for gigabit networks looks bright, as in fiber optics. ATM and FiberChannel are coming. As RAID subsystems become more popular, the idea of gigabit **ftp** sessions becomes closer to reality.

Acknowledgements

Thanks goes to the Systems Support Division at FNOC for their support and advice, and to the support and development staffs at Cray Research and CNT. I should point out that my name was already a four-letter word before I started working on the high-speed LAN.

Author Information

Jeff Pack is a system analyst with Grumman Data Systems at the Fleet Numerical Oceanography Center in Monterey, CA. He works on workstation and supercomputer administration, in addition to administering the high-speed LAN and can be reached at jpack@fnoc.navy.mil.

References

Clark, David D., *The Art and Engineering of Protocol Performance,* Interop Tutorial, August 1993.

Partridge, Craig, *Gigabit Network Architectures,* Interop Tutorial, August 1993.

Ultra Network Technologies, *Network Operations Manual,* 1990.

Loukides, Mike, *System Performance Tuning,* O'Reilly & Associates, Sebastopol, CA, 1991.

Stern, Hal L., *SunOS 4.1 Performance Tuning,* Internet White Paper.

The USENIX Association

The UNIX and Advanced Computing Systems Professional and Technical Association

The USENIX Association is a not-for-profit membership organization of those individuals and institutions with an interest in UNIX and UNIX-like systems and, by extension, C++, X windows, and other programming tools. It is dedicated to:

- sharing ideas and experience relevant to UNIX or UNIX inspired and advanced computing systems,
- fostering innovation and communicating both research and technological developments,
- providing a neutral forum for the exercise of critical thought and airing of technical issues.

Founded in 1975, USENIX sponsors twice yearly general conferences accompanied by vendor displays and frequent single-topic conferences and symposia. USENIX publishes proceedings of its meetings, the bi-monthly newsletter ;login:, the refereed technical quarterly Computing Systems. (published with the University of California Press), and is expanding its publishing role in cooperation with MIT Press with a book series on advanced computing systems. The Association actively participates in various ANSI, IEEE and ISO standards efforts with a paid representative attending selected meetings News of standards efforts and reports of many meetings are reported in ;login:.

SAGE, the Systems Administrators' Guild

USENIX has recently launched its first Special Technical Groups (STGs), the Systems Administrators' Guild (SAGE). SAGE is devoted to the advancement of systems administration as a profession. It will recruit talented individuals to the profession, develop guidelines for the education of members of the profession, establish standards of professional excellence and provide recognition for those who attain them, and promote work that advances the state of the art and propagates knowledge of good practice in the profession.

USENIX and SAGE will work together to publish technical information and sponsor conferences, symposia , tutorials and local groups in the field of systems administration. Currently USENIX and SAGE jointly sponsor the annual Systems Administration Conference and they, together with FedUNIX, are sponsoring the 1993 World Conference on Tools and Techniques for Systems Administration, Networking and Security (SANS-II). SAGE News and other items of interest to systems administrators are found in each issue of the USENIX newsletter ;login:.

There are four classes of membership in the Association, differentiated primarily by the fees paid and services provided.

USENIX Association services include:

- Subscription to login:, a bi-monthly newsletter;
- Computing Systems, a refereed technical quarterly;
- Discounts on various UNIX and technical publications for purchase;
- Technical conference and tutorial program twice a year and single-topic symposia periodically;
- A discount on technical conference and workshop registration fees;
- The right to vote on matters affecting the Association, its bylaws, election of its directors and officers.
- Right to join Special Technical Groups such as SAGE

The supporting members of the USENIX Association are:

ASANTÉ Technologies, Inc.
ANDATACO
Frame Technology, Inc.
Matsushita Electrical Industrial Co., Ltd.
Network Computing Devices, Inc.

OTA Limited Partnership
Quality Micro Systems
Sybase, Inc.
UNIX System Laboratories, Inc.
UUNET Technologies, Inc.

For further information about membership, conferences or publications, contact:

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Telephone: 510/528-8649
Email: office@usenix.org
FAX: 510/548-5738